Supervised Machine Learning Models' Performance Comparison

Md. Shafayat Hossain
Department of Computer Science and Engineering
East West University
2019-3-60-034@std.ewubd.edu

Rizvee Hassan Prito
Department of Computer Science and Engineering
East West University
2019-3-60-041@std.ewubd.edu

Md. Golam Rabiul Alam
Department of Computer Science and Engineering
East West University
golam.rabiul@ewubd.edu

Abstract— **The field of Machine Learning is very popular and widely used in different aspects. It is very popular in the field of predicting classification and regression problems. This study focused on different Supervised classification models such as SVM, KNN, Adaboost, Random Forest, Voting Classifier and Deep Neural Networks with and without batch processing to compare them and find the best performing model. The models have been used on the hotel reservation dataset which has 36275 instances and 19 features, only to do the comparison. The performance of the models heavily depends on different pre-processing techniques and the types of models used. The finding of this study is Deep neural networks(DNN) show the best performance.**

Keywords— Machine Learning, Supervised Learning, Classification, Comparison, KNN, SVM, AdaBoost, RF, VC, DNN.

I. Introduction

In the field of Artificial intelligence machine learning is one of the primary leading areas. Through the use of machine learning, one can train a machine to learn patterns in data and analyze it, furthermore to learn from it and adapt to it.

Machine learning is used in many areas because of its several applications such as classification and regression. Because of the vast diversity and unusual types of data, machine learning is further divided into supervised and unsupervised learning.

This study is focused on supervised classifications where the learner has to predict the output class and its pattern depending on the given input and output. This study will be conducting a comparison between supervised classification models. The comparison will be done on a few well-known supervised learning models. They are KNN, SVM, Adaboost, Random Forest, Voting Classifier, and Deep neural networks.

One of the simplest machine learning algorithms, based on the supervised learning method, is K-Nearest Neighbour. The K-NN algorithm makes the assumption that the new case and the existing cases are similar, and it places the new instance in the category that is most like the existing categories.

Support Vector Machine, or SVM, is one of the most well-liked supervised learning algorithms. It is used to solve Classification and Regression problems. The SVM algorithm's objective is to establish the best line or decision boundary that can divide n-dimensional space into classes, allowing us to quickly classify new data points. This best decision boundary is called a hyperplane.

AdaBoost, also known as Adaptive Boosting, is a machine learning method used as an ensemble method. Decision trees with one level are the most popular algorithm used with AdaBoost. It creates a model and equally weights each piece of data. Then, it gives larger weights to points that were incorrectly categorized. All of the points with greater weights are assigned more significance in the next model. It will continue to train models until a smaller error is observed.

As its name suggests, a random forest is made up of numerous independent decision trees that work together as an ensemble. Each tree in the random forest generates a class prediction, and the class with the most votes is the correct prediction.

A voting classifier is a machine learning model that receives training from an ensemble of various base models and predicts an output based on the output class with the highest likelihood.

Deep neural network is a neural network having at least two layers and some amount of complexity. It uses sophisticated math models to process data in complex ways.

To see which model is good for predicting, a hotel reservation dataset will be used to predict whether a hotel reservation will get canceled or not. People often book a hotel booking and later on for some circumstances cancel it. So it is better if it is possible to predict if a customer will cancel their booking or not in advance to lower the loss.

II. Methodology

This section provides details about how we used different supervised machine learning models for training and prediction, and the details of the dataset and dataset preprocessing techniques used in this research.

A. DATASET

The hotel reservation dataset is obtained from the Kaggle website. Its size is 3.08 MB and has 36275 rows and 19 columns. These are the columns of the dataset:

#	Column	Non-Null Count	Dtype
0	Booking_ID	36275 non-null	object
1	no_of_adults	36275 non-null	int64
2	no_of_children	36275 non-null	int64
3	no_of_weekend_nights	36275 non-null	int64
4	no_of_week_nights	36275 non-null	int64
5	type_of_meal_plan	36275 non-null	object
6	required_car_parking_space	36275 non-null	int64
7	room_type_reserved	36275 non-null	object
8	lead_time	36275 non-null	int64
9	arrival_year	36275 non-null	int64
10	arrival_month	36275 non-null	int64
11	arrival_date	36275 non-null	int64
12	market_segment_type	36275 non-null	object
13	repeated_guest	36275 non-null	int64
14	no_of_previous_cancellations	36275 non-null	int64
15	no_of_previous_bookings_not_canceled	36275 non-null	int64
16	avg_price_per_room	36275 non-null	float64
17	no_of_special_requests	36275 non-null	int64
18	booking_status	36275 non-null	object

From the above information, we can see that all the columns of the dataset have non-null values in all 36275 rows. All the columns have integer-type values except 5 of them have object-type values which are basically string values and one column has float-type values. The 'booking_status' column is the target\class\label column that is used to predict and find the models' accuracies. Other columns are feature columns. The 'booking_status' column has 2 unique values which are 'Canceled' and 'Not_Canceled' and it has 11885 'Canceled' values and 24390 'Not_Canceled' values in the dataset.

B. DATA PREPROCESSING

Data preprocessing is an important step that should be taken before the data is used to train the ml models. It improves the models' performance and helps to predict the class labels more accurately. For the preprocessing- first, all the unnecessary columns ('Booking_ID', 'arrival_year', 'arrival_date') for model training are dropped. Then, to drop the strongly correlated features\columns to reduce the features' dimension and model's training complexity, a heatmap of feature correlation is generated.

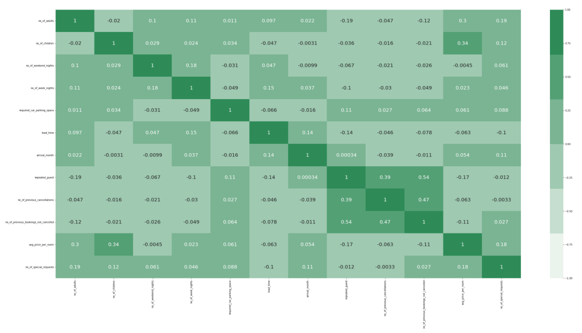


Figure 1: Features' correlation heatmap

From the heatmap, we can see that no feature is heavily correlated. So, no column is dropped for being strongly correlated with other columns. Then, 10477 duplicate rows are dropped to get better prediction accuracy of the models. Then, the categorical values are encoded with numbers so that machine learning models can use those values for training themselves in a better way. Then, standardScaler is used to remove the high variance and different scales of the features. It feature-wise transforms values in a way that the mean becomes zero and the standard deviation becomes one of every feature. High variance features badly affect the models' performance. Then, the principle component analysis (PCA) technique is used to simplify the complexity of data, reduce noise, and find latent features. Most importantly it is used to reduce features' dimensions and convert the values of correlated features in a way that, it will convert the features to linearly uncorrelated. If any feature's correlation scores with other features are more than 0.3 then the PCA's method of transforming features into uncorrelated features helps the model to improve its performance. In our obtained dataset, 4 features have more than a 0.3 correlation score. So, PCA is applied to transform these features linearly uncorrelated.

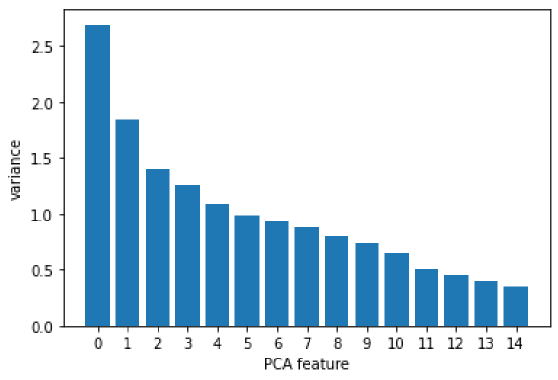


Figure 2: PCA implemented features' variance plot

Lastly, the whole dataset is split for models' training and testing. 80% of the dataset is used for training and 20% of the dataset is used for testing the models' prediction accuracy on unseen data. Besides these, one hot encoding is used on the label column to train the deep neural network.

C. MODELS' PARAMETERS:

Six different supervised machine learning models are implemented on the dataset. Structures of these models are determined through the grid search cross validation technique (GridSearchCV). GridSearchCV is a hyperparameter tuning process which is used to find the optimal parameters of models through a grid where some set of parameters has to be initialized. In the GridSearchCV, five-fold cross-validation is applied. This cross-validation is implemented on the entire dataset.

Six supervised ML models are K-Nearest Neighbor (KNN), Adaboost classifier (ADB), Support Vector Machine (SVM), Voting Classifier (VC), Random Forest classifier (RF), and Deep Neural Network(DNN).

For the KNN model, the n_neighbors parameter is used in the GridSearchCV technique. This parameter is used for selecting the minimum number of the nearest class's points for a new point to consider that new point as a member of that class.

For the AdaBoost classifier, n_estimator and learning rate are used for grid searching cross validation. N_estimator is the maximum number of estimators at which boosting is terminated. These estimators are also called stumps which are weak learners. Learning rate controls the loss function used for calculating the weight of the base models.

For the SVM model, Regularization parameter C and kernel are used in the GridSearchCV technique. The strength of regularization is inversely proportional to the value of C. Kernel parameters use mathematical equations to transform the data in a way that a linear classifier can solve a nonlinear problem caused by those data.

A voting classifier is a machine learning model that aggregates the results from several base models or estimators to make predictions. In the implemented Voting Classifier model, three base models are used which are KNN, SVM, and Decision tree. Grid search is implemented on the decision tree model of the voting classifier. The criterion parameter is used for grid searching cross validation. The measurement method for a split's impurity is determined by this parameter. The resulting parameters' values of previously implemented grid searching cross-validation on KNN and SVM are used in the KNN and SVM base models of the Voting Classifier.

For the Random Forest classifier, n_estimator and criterion parameters are used for grid searching cross-validation. The n_estimator parameter works in the same way as the n_estimator of the Adaboost classifier but instead of using stumps Random forest classifier uses weak learner trees which makes the Random forest a strong learner. The criterion parameter also works in the same way as the criterion parameter of the Decision tree classifier.

For the deep neural network model without batch input, epochs are used as the parameters for implementing the GridSearchCV technique. This parameter is used to select the number of times a model should use the entire dataset for training. And for the deep neural network model with batch input, batch_size and epochs are used as the parameters for implementing the GridSearchCV technique. The batch_size parameter determines how many samples from the dataset should be taken at once to train the model in each epoch.

Default parameters are used without using the GridSearchCV techniques.

Table 1: GridSearchCV's best parameters' and default parameters' values of the implemented models

Models	**Set of given parameters for GridSearch CV**	**GridSearchCV's Best Parameters**	**Default Parameters**
KNN	n_neighbors =[5,10,15,20, 25]	n_neighbors= 15	weights='uniform', algorithm='auto', leaf_size=30,

			metric=‘minkowski’, p=2
ADB	n_estimators = [100, 250, 500], learning_rate = [0.01, 0.1, 1.0]	n_estimators= 500, learning_rate= 1.0	algorithm=‘SAMME.R’
SVM	C=[50,20,10, 1], kernel= [‘linear’, 'poly', 'rbf', 'sigmoid']	C=50 kernel=‘rbf’	degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False
Voting Classifier (KNN, SVM, Decision Tree)	For Decision Tree, criterion=['gini', 'entropy', 'log_loss']	For Decision Tree, criterion= 'gini'	For Decision Tree, splitter='best', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, min_impurity_decrease=0.0, ccp_alpha=0.0
RF	n_estimator= [50,100,250, 500], criterion=['gini', 'entropy', 'log_loss']	N_estimator =50, criterion= 'gini'	min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', min_impurity_decrease=0.0, bootstrap=True, oob_score=False,verbose=0, warm_start=False,ccp_alpha=0.0,
DNN without Batch input	epochs=[30, 40,50,60,100]	epochs=30	activation='relu', activation='softmax', loss ='binary_crossentropy', optimizer='adam',metrics=['accuracy’]
DNN with Batch input	epochs=[30, 40,50,60,100], batch_size=[32,64,128,413]	epochs=30, batch_size=64	activation='relu', activation='softmax', loss= 'binary_crossentropy', optimizer='adam',metrics=['accuracy’]

D. Deep Neural Network Structure

```
Layer (type)          Output Shape        Param #
=====================================
dense_0 (Dense)        (None, 128)         2048

 dense_1 (Dense)       (None, 64)          8256

 dense_2 (Dense)       (None, 32)          2080

 dense_3 (Dense)       (None, 2)            66

=====================================
===========================

Total params: 12,450
Trainable params: 12,450
Non-trainable params: 0
=====================================
```

Our implemented deep neural network model has one input layer, three hidden layers, and one output layer. The input layer has 15 neurons or perceptrons because the dataset has 15 features for training. 128 neurons are used in the first hidden layer, 64 neurons are used in the second hidden layer and 32 neurons are used in the third hidden layer. The output layer has 2 neurons because the dataset has 2 unique class labels. Every hidden layer used ‘relu’ as the activation function and the output layer used ‘softmax’ as the activation function. The ‘binary_crossentropy’ is used as a loss function for the binary classification model. The ‘adam’ optimizer stands for Adaptive Moment Estimation which is an optimization technique for gradient descent. It works with the combination of the ‘gradient descent with momentum’ algorithm and the ‘RMSP’ algorithm.

III. Experimental Result and Discussion

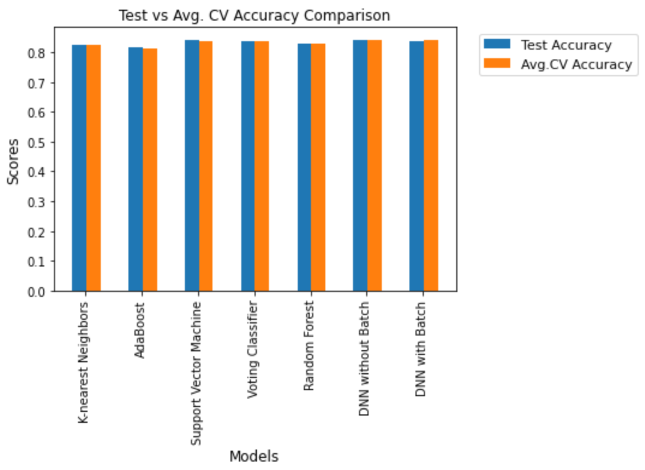


Figure 3: Models’ Test Accuracy and Average Cross Validation Accuracy scores comparison plot

Table 2: Models’ Test Accuracy scores and Average Cross Validation Accuracy scores comparison table:

Models	**Test Accuracy Scores**	**Average Cross Validation Accuracy Scores**
KNN	0.82403	0.82378
ADB	0.81550	0.81172
SVM	0.84089	0.83835
VC	0.83604	0.83549
RF	0.83003	0.83087
DNN without batch input	0.84089	0.84080
DNN with batch input	0.83934	0.84320

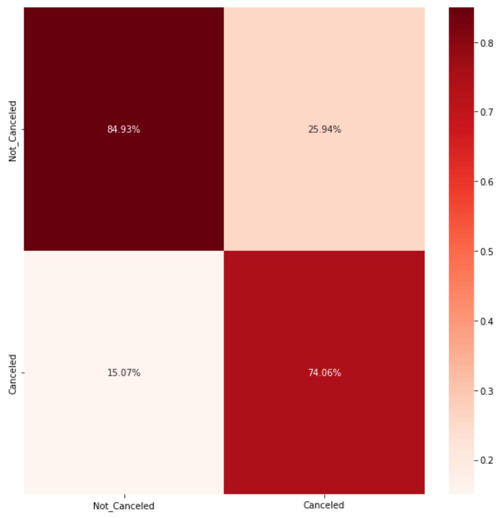
From the above table we see that the deep neural network has the highest accuracy score in both test accuracy scores and average cross-validation accuracy scores. Since, the deep neural network model uses high numbers of parameters/weights and tunes those weights in the training period, as a result, it predicts the class labels more accurately.

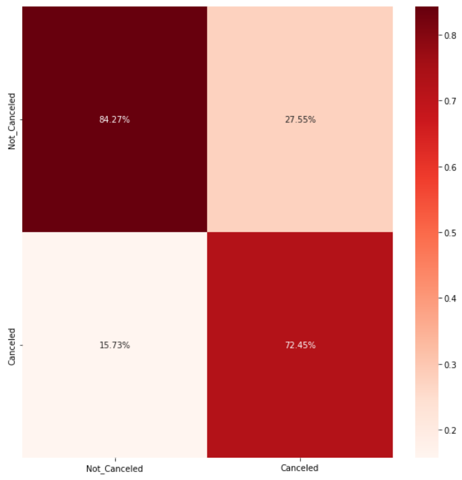
The deep neural network model that is trained with batch processing has lower testing accuracy and higher cross-validation accuracy than the DNN model trained without batch processing. Because at the time of measuring testing accuracy, the DNN with batch processing is trained on 80% data of the whole dataset. But at the time of applying cross-validation, the whole dataset is used. So, the model got more data in each batch for training and tuning the models’ hyperparameters for each batch of input. DNN upgrades its weights after each batch when batch processing is used and it upgrades its weights after each sample when batch processing is not used. So, because of the lack of data in batch, DNN trained without batch processing performed better than the DNN in the period of measuring testing accuracy on testing data.

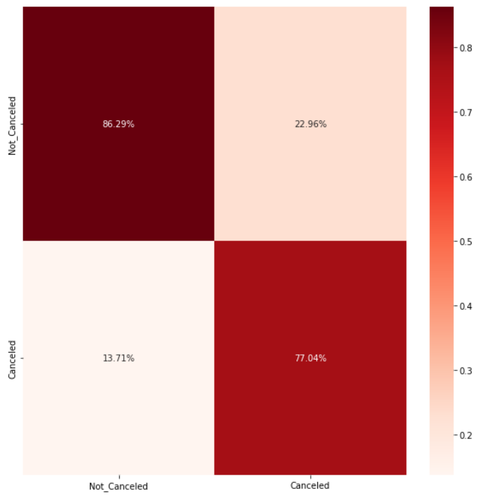
The second best model is SVM in terms of both testing and cross-validation accuracy. It has a higher accuracy score in both cases because a high value is used in the regularization parameter. Since, the regularization parameter is inversely proportional to the strength of regularization so, using the high value in this parameter has given the lowest penalty to the SVM model. Regularization is used for making the model less overfit. Since our dataset is not too small, the SVM model didn’t get overfit rather, it was facing some underfitting problems. So, after using low regularization it has fitted more with the datasets and performed better accuracy scores in prediction.

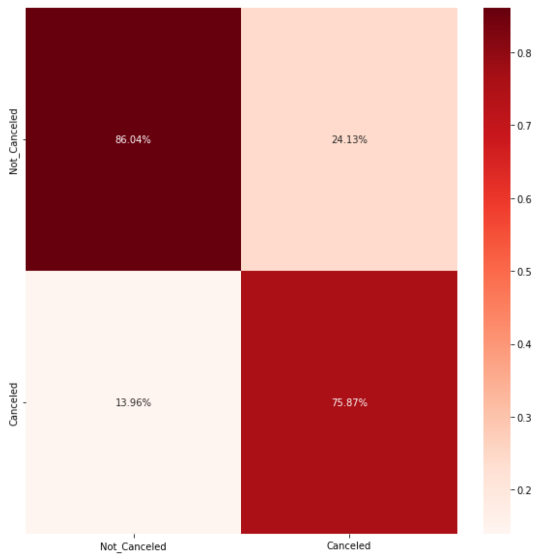
Adaboost has given the lowest accuracy scores in terms of test accuracy and cross-validation accuracy among all the implemented models. Because the AdaBoost classifier uses stumps which are weak learners and have one node and two leaves. Random Forest classifier also uses weak learners as trees but to make those trees more than one feature is considered. Since AdaBoost only considers one feature while making the stump, therefore, stumps are not great at making accurate classifications.

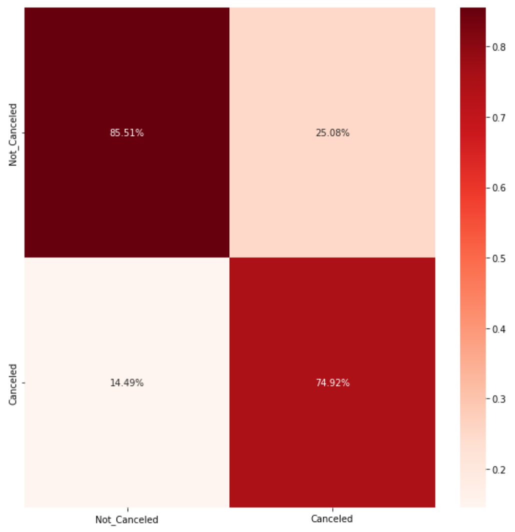
Heatmap of the confusion matrix of KNN, ADB, SVM, VC, RF, DNN without batch processing and DNN with batch processing are given below respectively:

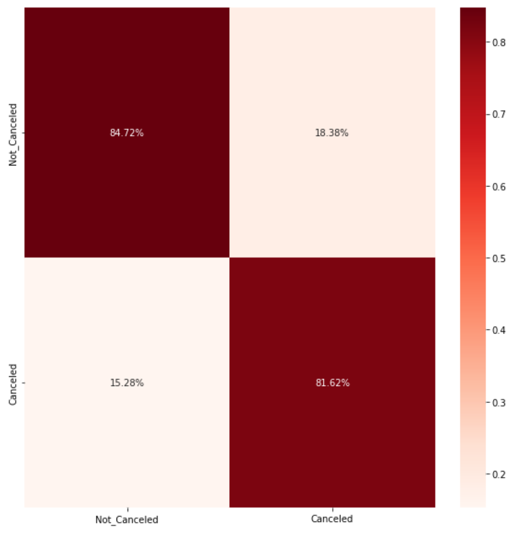


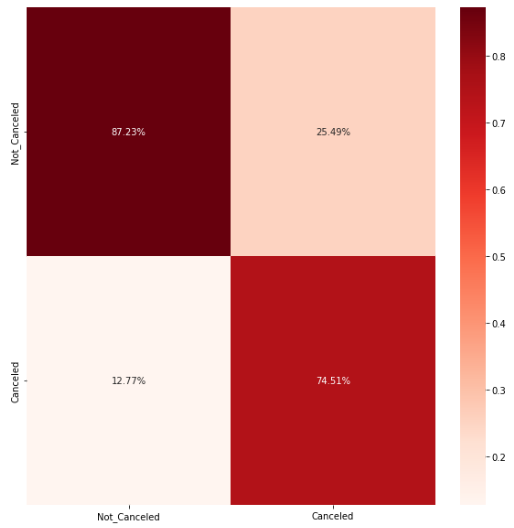












From these heatmaps, we can see that the 'Canceled' label is less accurately predicted than the 'Not_Canceled' label by all the models. Because in the dataset, the number of samples of the 'Canceled' label is very lower than the other class label. That is why models couldn't fit appropriately to the samples of the 'Canceled' label. As a result, models predicted the 'Canceled' label more wrongly.

IV. Conclusion & Future Works

To have good classification model prediction, the models need careful fine-tuning of the models' parameters with a large number of data instances. Also, building the model for the algorithm requires precision and accurate categorization, not just time.

In this study, we have compared six supervised classification models. Though in our case we have seen that the best result is given by Deep Neural Networks models. But this can not be for all cases because datasets are always vastly diverse and unusual. So different classification models can perform better for different datasets.

Here, this study has only conducted the performance comparison on one type of dataset. In the future, the comparison can be conducted on different types of datasets to measure the performance to a higher degree.

Colab Link of the code:

https://colab.research.google.com/drive/1W3eQcLI_j4tSCR5qHYZLdSfYoDPwJZds?usp=sharing